Docket No. 13346US01

CERTIFICATE OF MAILING

Express Mail Mailing Label No. EL849006495US
Date of Deposit February 28, 2002

I hereby certify that this paper or fee is being deposited with the United States Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to the Assistant Commissioner of Patents and Trademarks, Washington, DC 20231.

Mailer Ronald E. Larson
(print)
Mailer Ronald E Larson
(signature)

COMPRESSED AUDIO STREAM DATA DECODER MEMORY SHARING TECHNIQUES

BACKGROUND OF THE INVENTION

[0001] This invention relates to decoding of compressed audio stream data and particularly relates to allocating memory for the decoding of such data.

[0002] Decoding compressed audio stream data frequently involves removing jitter from the compressed data. Removing the jitter requires implementation of a jitter buffer, which stores an amount of compressed data, such as compressed packet voice data. Packets typically are put into the jitter buffer from a packet network at a non-

constant rate (i.e., with jitter). Data is extracted from the buffer at a constant rate and played out into the telephone network. Previous decoders have sized the jitter buffer to hold a given amount of G.711 data (pulse code modulated (PCM) data transmitted at 64 kilo bits per second(kbps)). This jitter buffer size is usually expressed in bytes or the time duration of G.711 samples. The jitter buffer consumes a significant amount of memory. For example, a 200 millisecond (ms) jitter buffer requires at least 1600 bytes of memory (i.e. 200ms of G.711 data requires 1600 bytes). Previous decoders do not have the means of re-sizing a jitter buffer whenever the voice decoder algorithm changes dynamically, based on the type of voice decoder algorithm in use. As a result, prior decoders have wasted memory and required excessively large memories. This invention addresses the problem and provides a solution.

[0003] Further limitations and disadvantages of conventional and traditional approaches will become apparent to one of skill in the art, through comparison of such systems with the present invention as set forth in the

remainder of the present application with reference to the drawings.

BRIEF SUMMARY OF THE INVENTION

[0004] One apparatus embodiment of the invention is useful in a decoder for decoding compressed audio stream data. In such an environment, decoding apparatus comprises a memory arranged to store the compressed data and to store at least one of operating data and operating code for a plurality of decompression algorithms requiring different amounts of memory for the operating data and operating code and requiring different amounts of memory to store compressed data corresponding to a predetermined duration of uncompressed data. A processor is arranged to select one of the decompression algorithms, to allocate an amount of the memory for storing compressed data and at least one of operating data and operating code depending on the decompression algorithm selected and to decode the compressed data stored in the allocated amount of memory.

[0005] One method embodiment of the invention is useful for allocating memory for decoding compressed audio stream

data. In such an environment, the memory is allocated by steps comprising storing at least one of operating data and operating code for a plurality of decompression algorithms requiring different amounts of memory for the operating data and operating code and requiring different amounts of memory to store compressed data corresponding to a predetermined amount of uncompressed data. One of the decompression algorithms is selected and an amount of the memory is allocated for storing compressed data and at least one of operating data and operating code depending on the decompression algorithm selected. At last a portion of the compressed data is stored in the allocated amount of memory, and the stored compressed data is decoded using the selected decompression algorithm.

[0006] Another embodiment of the invention is useful in a computer readable media encoded with executable instructions representing a computer program that can cause a computer to dynamically size memory by performing the tasks of storing at least one of operating data and operating code for a plurality of decompression algorithms requiring different amounts of memory for the operating

data and operating code and requiring different amounts of memory to store compressed data corresponding to a predetermined amount of uncompressed data. One of the decompression algorithms is selected, and an amount of the memory is allocated for storing compressed data and at least one of operating data and operating code depending on the decompression algorithm selected. At last a portion of the compressed data is stored in the allocated amount of memory, and the stored compressed data is decoded using the selected decompression algorithm.

[0007] These and other advantages and novel features of the present invention, as well as details of an illustrated embodiment thereof, will be more fully understood from the following description and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] Figure 1 is a schematic block diagram of a one embodiment of the invention.

[0009] Figure 2 is a flow diagram illustrating one method embodiment of the invention and describing a portion of the operation of the apparatus shown in Figure 1.

DETAILED DESCRIPTION OF THE INVENTION

[0010] The inventor has recognized that for a range of voice decompression algorithms, the more highly compressed the data the more operating data plus operating code the algorithm consumes. Furthermore, a lower bit rate algorithm requires a smaller jitter buffer to store a given duration of data. Prior decoders allocated a worst-case amount of memory for voice decompression algorithm operating data (plus optionally operating code) plus a worst-case amount of memory for a jitter buffer. The inventor recognized that both worst cases do not occur simultaneously. The embodiments described in this specification provide for dynamic resizing of a jitter buffer based on the voice decompression algorithm needed for decoding. Thus, the amount of memory allocated for decompression algorithm operating data (plus optionally operating code) plus compressed data can be changed depending on the algorithm used for decoding. One example of dynamic memory allocation is allocation during a phone call depending on the type of compressed data that needs to be decoded.

[0011] The embodiments of a decoder 10 made in accordance with the invention will be explained in the context of a voice over packet network (e.g., VoIP or VoATM) application. Referring to Figure 1, packets of G.711, G.726, G.728 or another voice encoder (vocoder) algorithm encoded (i.e., compressed) voice data are received on a channel 20. The packets typically are generated by a vocoder, which forms part of a voice transmitter.

[0012] A depacketizing engine 30 identifies the type of packets received from the host (i.e., voice packet, DTMF packet, call progress tone packet, SID packet, etc.), and transforms the packets into frames that are protocol independent. The depacketizing engine 30 then transfers the voice frames (or voice parameters in the case of silence identifier (SID) packets) into a voice decoder 40. (The depacketizing engine 30 also may transfer DTMF frames into a DTMF queue not shown and transfer call progress tones into the call progress tone queue not shown.)

[0013] Voice decoder 40 includes a central processing unit (CPU) 42 and a memory 44. Memory 44 is a computer readable media that stores either operating data and operating code or operating data without operating code for decompression algorithms corresponding to the G.711, G.726, G.728 or another vocoder algorithm encoded data. The packets of data include information identifying the type of encoded data (e.g., G.711, G.726 or G.728 encoded data). In response to this information, CPU 42 selects the appropriate algorithm required for decoding the data and allocates an amount of memory 44 for storing compressed data and either operating data and operating code or operating data without operating code for the selected algorithm. CPU 42 then stores either the operating data and operating code or operating data without operating code for the algorithm in memory 44 (if the operating data and operating code is not already located in memory 44) and stores compressed data from engine 30 in the allocated memory. For example, a portion 46 of memory 44 is used to store operating data and operating code for the selected

algorithm and a portion 48 of memory 44 is used to store compressed data from engine 30.

[0014] The allocation of memory may occur at the beginning of a phone call represented by compressed voice data received on channel 20. Thus, CPU 42 dynamically allocates memory 40 as needed depending on the type of compressed data used for the call and the type of decompression algorithm required to decode the data.

[0015] CPU 42 then removes jitter from the compressed data stored in memory portion 48, decodes the compressed data from memory portion 48 and transmits the resulting decompressed data to a media queue 60. Queue 60 may be used for various purposes, but is not needed for all applications. For example, a tone generator may overwrite queue 60 to generate DTMF tones. The data from the queue 60 is transmitted to a switch board 70 and then to a physical device (PXD) 80, which provides two way communication with a telephone or a circuit-switched network, such as a PSTN line (e.g. DS0) carrying a 64kb/s

pulse code modulated (PCM) signal, i.e., digital voice samples.

[0016] CPU 42 includes a program memory 43, which stores computer code representing an algorithm by which the memory allocating functions described in this specification are performed. This computer code may include the operating code for the decompression algorithms. Note that some implementations may execute the decompression algorithm operating code directly from program memory 43 or may instead copy operating code for a specific decompression algorithm to memory 44 and execute it directly from memory 44. Those skilled in communications and programming are able to write such code from the functions described in this specification.

[0017] Referring to Figure 2, the decoder shown in Figure 1 operates to allocate memory in the following manner. In step S100, CPU 42 stores either operating data and operating code or operating data without operating code for decompression algorithms of the types previously described. In step S102, compressed data of the type

previously described is received from channel 20 at the beginning of a call. In step S104, information in the data is analyzed by CPU 42 to determine the type of compression used for the call, and an appropriate one of the decompression algorithms is selected. In step 106, an appropriate amount of memory 44 is allocated for compressed data and for either operating data and operating code or operating data without operating code for the selected decompression algorithm. In step S108, either operating data and operating code or operating data without operating code is stored in memory portion 46 and compressed data is stored in memory portion 48. In step S110, the compressed data stored in memory portion 48 is decoded, including the removal of jitter by well-known algorithms.

[0018] The steps shown in Figure 2 are used to implement a gateway packet voice exchange service (PVE). The memory consumption of the PVE Service is affected by its compilation parameters. There are two main factors affecting memory 44 consumption:

(1) The amount of either operating data and operating code or operating data without operating code requiring storage in portion 46 of memory 44 varies with the combination of vocoders chosen via compilation parameters. More complex vocoders generally require more operating data and operating code.

(2) The amount of compressed data requiring storage in portion 48 of memory 44 varies with a parameter, such as XCFG_GLOBAL_MAX_JITTER_MSEC, which represents the maximum duration of audio stream signal (e.g., voice) that can be generated from the compressed data. Compressed data of greater complexity generally can generate a longer duration of voice signal and sound with fewer stored bytes.

[0019] Since jitter buffer duration is expressed in milliseconds (ms), the actual consumption in bytes of compressed data is dependent upon the type of compressed data stored within portion 48 of memory 44. For example, 200 ms of G.711 encoded data requires 1600 bytes, whereas 200 ms of G.728 encoded data requires 400 bytes. Each

vocoder consumes a different amount of memory while it is running. Generally more complex vocoders consume more operational data and operational code. However the more complex vocoders have a lower bit rate and therefore need less coded data in memory portion 48 to generate the same duration of audio stream signal. The PVE Service exploits this trade-off between vocoder decoder operational memory (i.e. instance memory) and compressed data memory to reduce its overall memory consumption. For an example, where operational code is not stored in memory 44, if the PVE Service supports G.711, G.726 and G.728 encoded data, CPU 42 allocates memory 44 according to the expression:

```
PVE_memory = MAXIMUM (
     G711d_inst + XCFG_GLOBAL_MAX_JITTER_MSEC*8,
     G726d_inst + XCFG_GLOBAL_MAX_JITTER_MSEC*5,
     G728D_inst + XCFG_GLOBAL_MAX_JITTER_MSEC*2  )
```

Where Gxxxd_inst represents the amount of decoder operational data stored in memory portion 46 for the G.xxx vocoder, XCFG_GLOBAL_MAX_JITTER_MSEC*x represents the amount of compressed data stored in memory portion 48 for the G.xxx vocoder, and PVE_memory represents the total

amount of memory allocated in memory 44 for the selected vocoder.

[0020] The foregoing method of resizing memory 44 based on the decompression algorithm type saves memory compared to statically sizing the memory 44 for the worst case memory requirement. For statically sizing a memory able to hold 200 ms of G.711 encoder data;

```
Required memory 44 amount =
    MAXIMUM (G711d_inst, G726d_inst, G728_inst) +
    XCFG_GLOBAL_MAX_JITTER_MSEC*8
```

Typical values for the above example are:

```
G711d_inst = 16 bytes
G726d_inst = 68 bytes
G728d_inst = 2020 bytes
XCFG_GLOBAL_MAX_JITTER_MSEC = 200
```

Thus the memory saving by not using a static memory equals 1200 bytes (i.e., 1600 bytes required for G.711 minus 400 bytes required for G.728).

[0021] Although memory portions 46 and 48 have been shown as distinct and separate in Figure 1, those skilled

in the art will recognize that the memory portions could be arranged in other ways, such as interleaved.

[0022] While the invention has been described with reference to one or more preferred embodiments, those skilled in the art will understand that changes may be made and equivalents may be substituted without departing from the scope of the invention. For example, although only a single channel is shown in Figure 1, the invention can be used with multiple channels. Typically, a chip used to implement the invention would service hundreds of channels. As a result, the amount of memory saved by such a chip is substantial. The depacketizing engine and decoder shown in Figure 1 may be repeated for each channel, or depending on the size of memory 44 and speed of CPU 42, a single memory and CPU may be used for more than one channel.

[0023] In addition to memory 44, computer readable media within the scope of the invention include magnetic media, such as floppy disks and hard drive, as well as optical media, including CD-ROMs and DVDs.

[0024] In addition, many modifications may be made to adapt a particular step, structure, or material to the teachings of the invention without departing from its scope. Therefore, it is intended that the invention not be limited to the particular embodiment disclosed, but that the invention will include all embodiments falling within the scope of the appended claims.